

Лабораторная работа №1

Создание контейнерного приложения в IBM Cloud.

Требования:


Наличие аккаунта в IBM Cloud.

Установленные на локальной машине – Git, Docker, текстовый редактор.

Этап 1. Создание кластера Kubernetes в IBM Cloud.

В данной лабораторной работе будет использован демо-кластер Kubernetes, расположенный в IBM Cloud.

Для создания кластера необходимо выполнить следующие шаги:

- Перейти по адресу <http://bluemix.net>
- Войти со своими учетными данными или создать новый аккаунт
- После успешного входа вы увидите следующий экран (рис.1).
- Необходимо раскрыть меню, нажав в левом верхнем углу на значок 

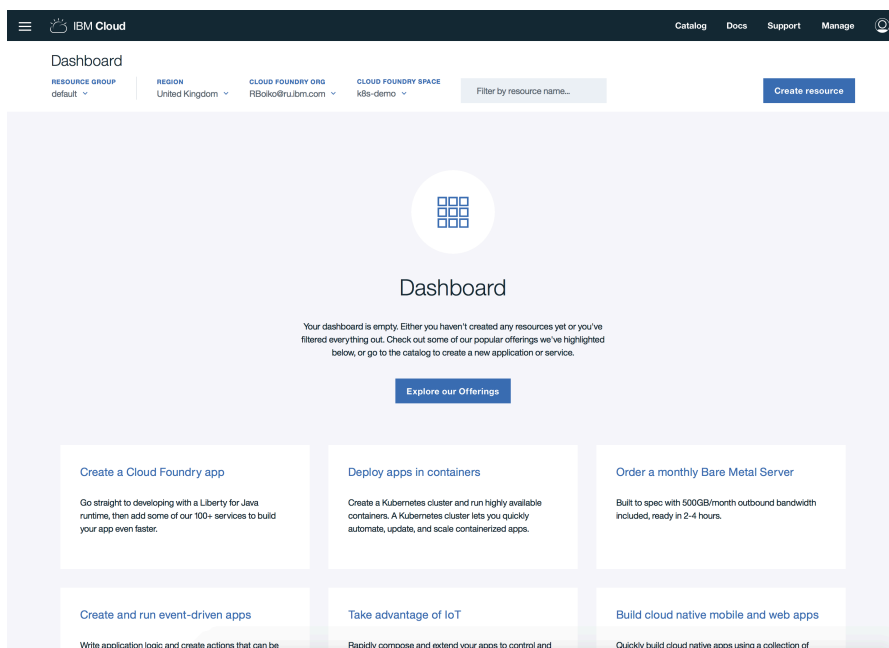


рисунок 1

- В раскрывшемся меню (рис. 2) необходимо выбрать опцию Containers

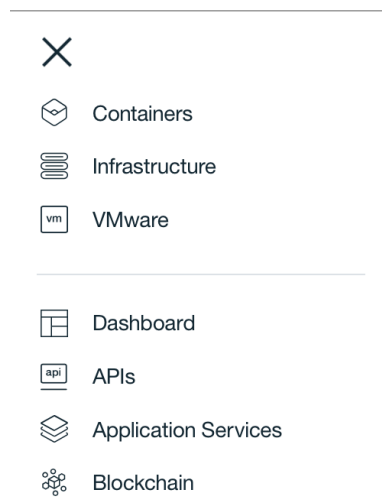


рисунок 2

- На открывшейся странице (рис.3) необходимо нажать на кнопку Create Cluster

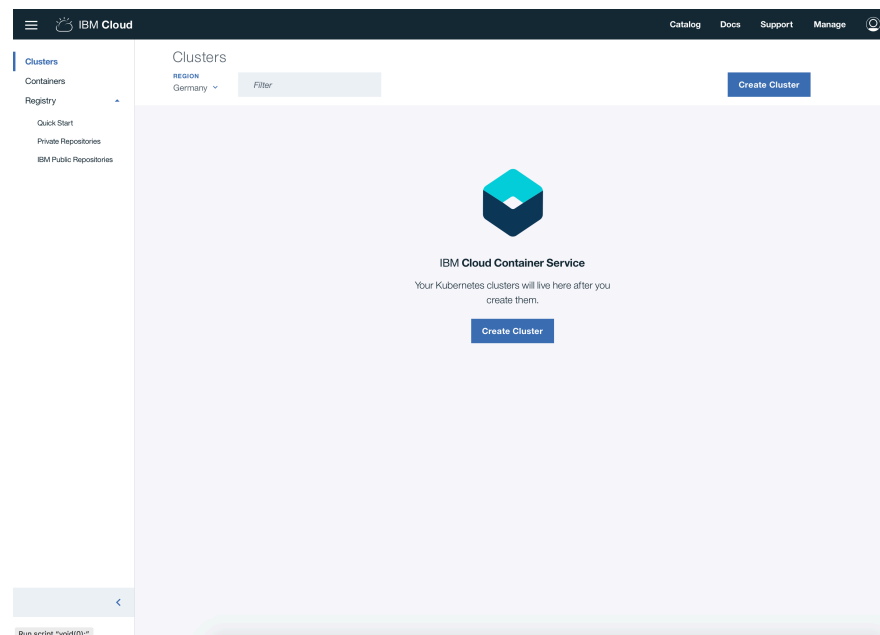


рисунок 3

- На странице создания кластера (рис. 4) необходимо убедиться, что выбран Cluster Type – Lite Plan. Остальные значения надо оставить по-умолчанию и нажать кнопку Create Cluster.

Dashboard / Clusters / Create Cluster

Provision a cluster of hosts, called worker nodes, to deploy and manage highly-available apps.

[Docs](#) [Terms](#)

Cluster type

Lite plan
 New to Kubernetes? Create a cluster with 1 worker node to explore the capabilities.
 Free

Pay-As-You-Go plan
 Create a fully-customizable, production-ready cluster with your choice of hardware.
 Starting from \$0.19 hourly

Cluster details

Cluster Name
mycluster

Location **Kubernetes version**

Machine type
 Lite
 2 CPUs, 4 GB RAM
 Free

Worker nodes
1

Private VLAN
Public VLAN

Order Summary

Lite - 2 CPUs, 4 GB RAM
1 worker node
Free

Total due now: Free estimated

[Create Cluster](#)
[Cancel](#)

рисунок 4

- Далее вы увидите страницу с описанием настроек для подключения (рис. 5), они понадобятся на следующих этапах.

Dashboard / Clusters / mycluster ● Deploying

Gain access to your cluster

Prerequisites
To gain access to your cluster, download and install a few CLI tools and the IBM Cloud Container Service plug-in.
[IBM Cloud CLI](#)
[Kubernetes CLI](#)

```
bx plugin install container-service --r Bluemix
```

Gain access to your cluster
Log in to your IBM Cloud account.

```
bx login -a https://api.eu-gb.bluemix.net
```

If you have a federated ID, use `bx login -reso` to log in to the IBM Cloud CLI.

Set your terminal context to your cluster.

```
bx cs cluster-config mycluster
```

In the output, the path to your configuration file is displayed as a command to set an environment variable, for example:

```
export KUBECONFIG=/Users/lhe/.bluemix/plugins/container-service/clusters/mycluster/kube-config-par01-mycluster.yml
```

Copy and paste the command to set the environment variable in your terminal and press Enter.
Alternatively, you may directly [download](#) your kubeconfig file to manually configure the Kubernetes cluster context.

You're all set. You can verify that you can connect to your cluster by listing the cluster's worker nodes.

```
kubectl get nodes
```

You can also access your Kubernetes dashboard.

```
kubectl proxy
```

рисунок 5

- Необходимо перейти на вкладку Overview, на открывшейся странице с деталями кластера (рис. 6) необходимо обратить внимание на поля Location и Managed From. Значения из этих полей потребуются на следующих шагах лабораторной работы.

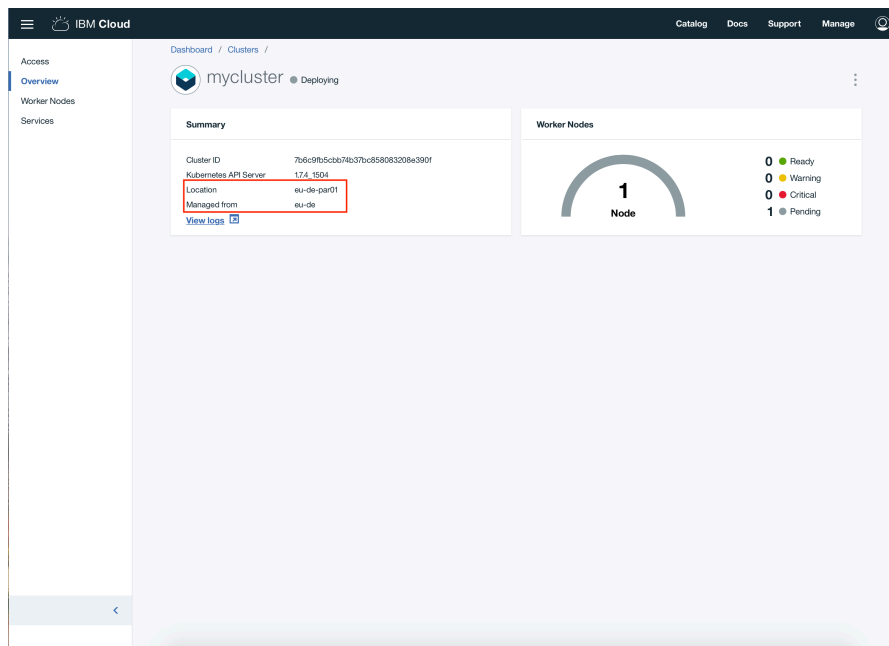


рисунок 6

- На этом первый этап завершен

Этап 2. Создание контейнерного приложения.

В рамках данного этапа будет создано простейшее Node.js приложение, связанное с NoSQL БД MongoDB. Для MongoDB, также будет запущен отдельный контейнер с инструментами администрирования. Данное приложение будет протестировано на локальном хосте.

Для создания и запуска приложения необходимо выполнить следующие шаги:

- Открыть терминальное окно
- Клонировать приложение командой
`git clone https://github.com/roman-boiko/demoapp.git`
- Перейти в папку с приложением
`cd demoapp`
- В папке надо создать файл с именем `Dockerfile` и следующим содержанием

```
FROM node:8.9.1
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["npm", "start"]
```

- Запустите сборку образа
`docker build -t demoapp .`
- Проверьте список образов
`docker image ls`
- Для запуска нескольких связанных контейнеров будем использовать утилиту `docker-compose`
- Установка `docker-compose`

```
sudo curl -L https://github.com/docker/compose/releases/download/1.17.0/docker-  
compose-`uname -s`-`uname -m`-o /usr/local/bin/docker-compose
```

```
sudo chmod +x /usr/local/bin/docker-compose
```

- Создайте файл `docker-compose.yml` со следующим содержимым

```
node:  
  build: .  
  links:  
    - mongo  
  ports:  
    - "3000:3000"  
  
mongo-express:  
  image: mongo-express:0.42.2  
  links:  
    - mongo  
  ports:  
    - "8081:8081"  
  
mongo:  
  image: mongo:3.4.10
```

- Запустите приложение
`docker-compose up`
- Приложение доступно по адресу <http://localhost:3000>
- Интерфейс администрирования MongoDB доступен по адресу <http://localhost:8081>
- Остановите локальную копию приложения
`docker-compose down`
- На этом второй этап завершен

Этап 3. Запуск приложения в IBM Cloud.

- Для работы с кластером потребуются инструменты командной строки – IBM Cloud CLI и kubectl
- Установка IBM Cloud CLI
`curl -fsSL https://clis.ng.bluemix.net/install/linux | sh`
- Установка дополнений к IBM Cloud CLI
`bx plugin install container-service -r Bluemix`
`bx plugin install container-registry -r Bluemix`
- Установка kubectl
`curl -LO https://storage.googleapis.com/kubernetes-
release/release/v1.7.3/bin/linux/amd64/kubectl`
`chmod +x ./kubectl`
`sudo mv ./kubectl /usr/local/bin/kubectl`

- Залогиньтесь в IBM Cloud, при входе надо будет указать тот регион в котором у вас был создан кластер, например, `eu-de`
`bx login`
- Создайте новое пространство имен в Docker реестре
`bx cr namespace-add <my_namespace>`
- Залогиньтесь в созданном пространстве
`bx cr login`
- Создайте метку для локального образа
`docker tag demoapp`
`registry.<my_region>.bluemix.net/<my_namespace>/<my_repository>:<my_tag>`
- Загрузите локальный образ в реестр
`docker push`
`registry.<my_region>.bluemix.net/<my_namespace>/<my_repository>:<my_tag>`
- Проверьте, что образ успешно загрузился
`bx cr image-list`
- Сконфигурируйте подключение к кластеру
`bx cs cluster-config mycluster`
- Проверьте подключение, выведя список воркер-узлов(рис. 7)
`kubectl get nodes`

```
ibm@containers-vm:~/demoapp$ kubectl get nodes
NAME                STATUS    AGE           VERSION
10.126.128.232      Ready     6h            v1.7.4-2+eb9172c211dc41
```

рисунок 7

- Создайте файл `demoapp.yml`, указав в нем ссылку на ваш реестр

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: demoapp
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: demoapp
    spec:
      containers:
        - name: node
          image: registry.<region>.bluemix.net/<namespace>/<repository>:<tag>
          ports:
            - containerPort: 3000
---
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: mongo
spec:
  replicas: 1
  template:
    metadata:
      labels:
```

```
    app: mongo
  spec:
    containers:
      - name: mongo
        image: mongo:3.4.10
        ports:
          - containerPort: 27017
---
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: mongo-express
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: mongo-express
    spec:
      containers:
        - name: mongo-express
          image: mongo-express:0.42.2
          ports:
            - containerPort: 8081
---
kind: Service
apiVersion: v1
metadata:
  name: mongo
spec:
  selector:
    app: mongo
  type: ClusterIP
  ports:
    - name: mongo
      port: 27017
      targetPort: 27017
---
kind: Service
apiVersion: v1
metadata:
  name: demoapp
spec:
  selector:
    app: demoapp
  type: LoadBalancer
  ports:
    - name: demoapp
      port: 3000
---
kind: Service
apiVersion: v1
metadata:
```

```

name: mongo-express
spec:
  selector:
    app: mongo-express
  type: LoadBalancer
  ports:
  - name: mongo-express
    port: 8081

```

- Запустите установку
`kubectl apply -f demoapp.yml`
- Проверьте, что установка прошла успешно
`kubectl get deploy` (рис. 8)

```

lbn@containers-vm:~/demoapp$ kubectl get deploy
NAME          DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
demoapp       1         1         1             1           7m
mongo         1         1         1             1           7m
mongo-express 1         1         1             1           7m

```

рисунок 8

`kubectl get po` (рис. 9)

```

lbn@containers-vm:~/demoapp$ kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
demoapp-4267556569-7449h           1/1     Running   1           6m
mongo-3012733415-0xb6c             1/1     Running   0           6m
mongo-express-278801216-jltx2      1/1     Running   1           6m

```

рисунок 9

- Получите список сервисов
`kubectl get svc` (рис. 10)

```

lbn@containers-vm:~/demoapp$ kubectl get svc
NAME          CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
demoapp       172.21.0.20     <pending>     3000:30157/TCP   8m
kubernetes    172.21.0.1      <none>        443/TCP          6h
mongo         172.21.145.36   <none>        27017/TCP        8m
mongo-express 172.21.62.151   <pending>     8081:30195/TCP   8m

```

рисунок 10

- Получите список узлов кластера
`bx cs workers mycluster` (рис. 11)

```

lbn@containers-vm:~/demoapp$ bx cs workers mycluster
OK
ID                                Public IP      Private IP      Machine Type  State  Status  Version
kube-par01-pa7b6c9fb5cbb74b37bc858083208e390f-w1 169.51.16.127 10.126.128.232 free          normal Ready    1.7.4_1504

```

рисунок 11

- Сервисы доступны по публичному IP адресу и советующему порту, например, в данном случае демоапп доступен по адресу <http://169.51.16.127:30157>, а mogo-express по адресу <http://169.51.16.127:30195>
- На этом третий этап и вся лабораторная работа успешно завершены.